Applied Deep Learning

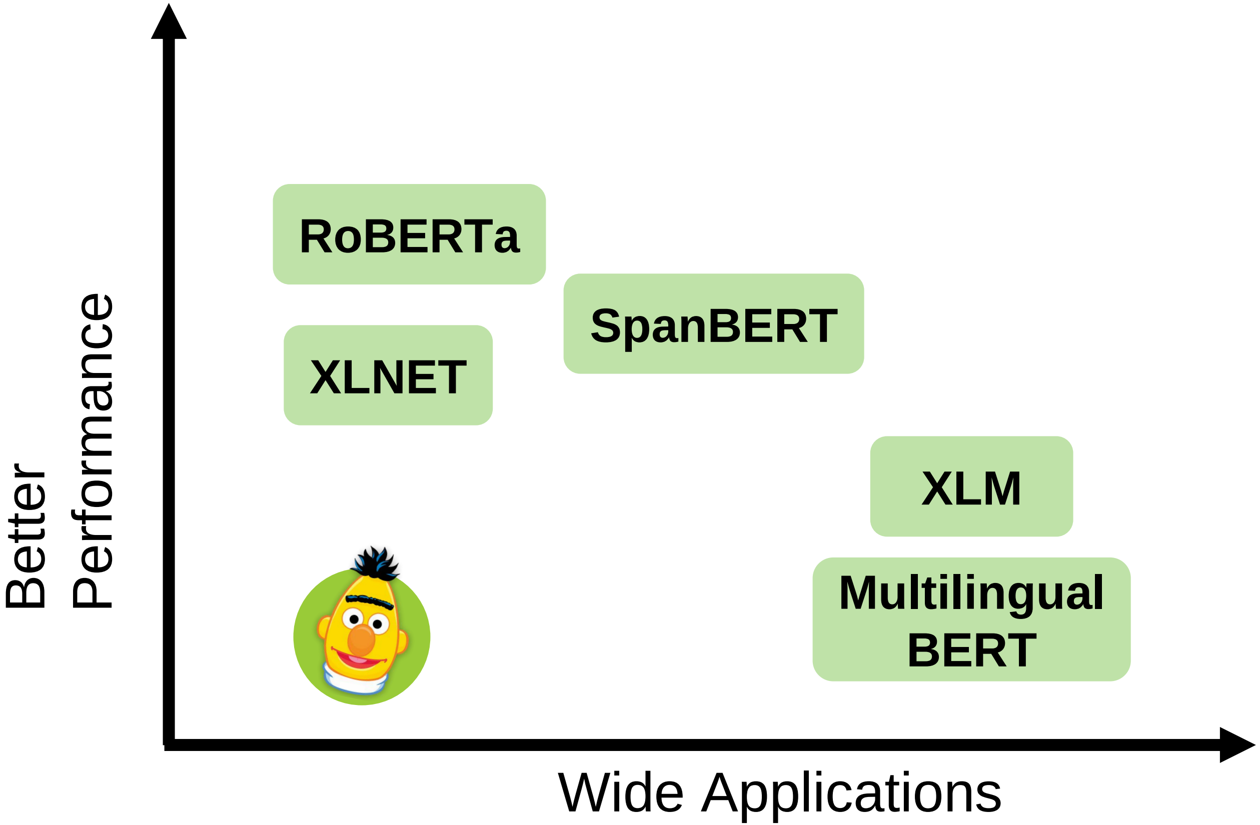
More BERT

March 28th, 2022 http://adl.miulab.tw

National Taiwan University
國立臺灣大學

2

Beyond BERT



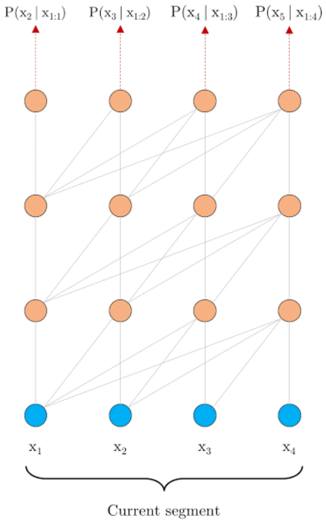
3

Transformer-XL

(Dai et al, 2019)

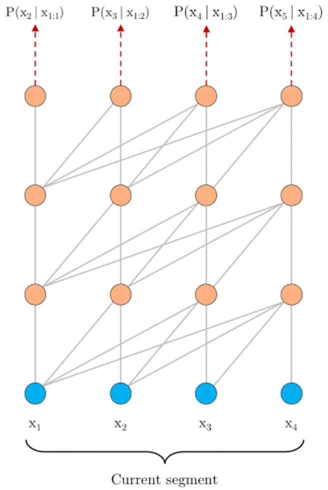
4 Transformer

- Issue: context fragmentation
  - Long dependency: unable to model dependencies longer than a fixed length
  - Inefficient optimization: ignore sentence boundaries
    - particularly troublesome even for short sequences



5 Transformer-XL (extra-long)

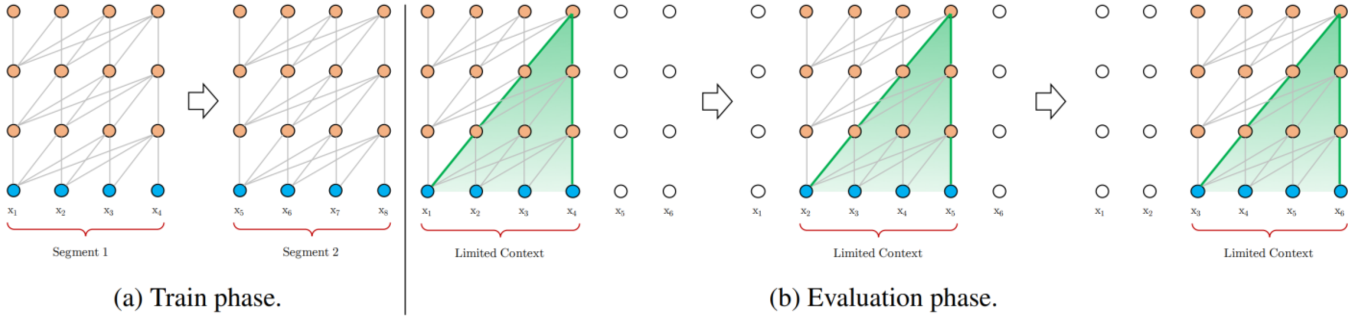
- Idea: segment-level recurrence
  - Previous segment embeddings are **fixed** and **cached** to be reused when training the next segment
  - → increases the largest dependency length by N times (N: network depth)



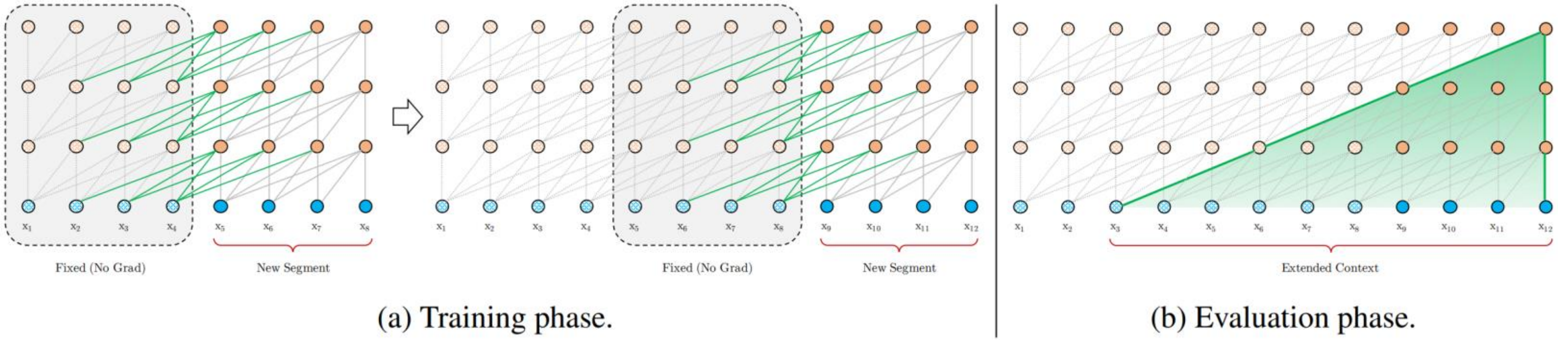
resolve the context fragmentation issue and makes the dependency longer

6 State Reuse for Segment-Level Recurrence

- Vanilla



- State Reuse



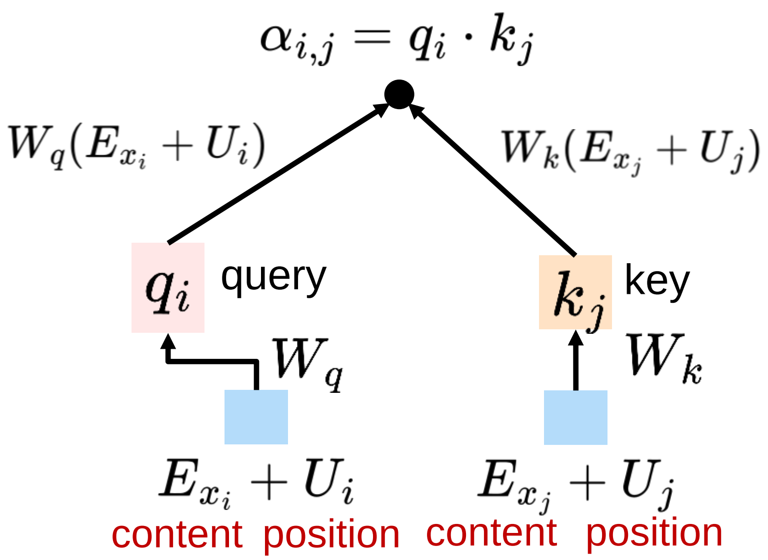
7

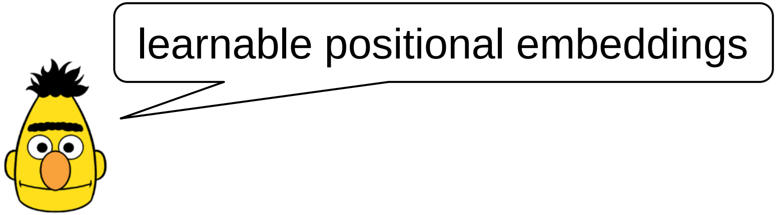
Incoherent Positional Encoding

- Issue: naively applying segment-level recurrence can't work
  - positional encodings are *incoherent* when reusing

$$[0, 1, 2, 3] \rightarrow [0, 1, 2, 3, 0, 1, 2, 3]$$

Self-Attention with Absolute Positions





$$\mathbf{A}_{i,j}^{\mathrm{abs}} = (\mathbf{E}_{x_i} + \mathbf{U}_i)^\top \mathbf{W}_q^\top \mathbf{W}_k (\mathbf{E}_{x_j} + \mathbf{U}_j)$$

$$\mathbf{A}_{i,j}^{\mathrm{abs}} = \mathbf{E}_{x_i}^\top \mathbf{W}_q^\top \mathbf{W}_k \mathbf{E}_{x_j} + \mathbf{E}_{x_i}^\top \mathbf{W}_q^\top \mathbf{W}_k \mathbf{U}_j + \mathbf{U}_i^\top \mathbf{W}_q^\top \mathbf{W}_k \mathbf{E}_{x_j} + \mathbf{U}_i^\top \mathbf{W}_q^\top \mathbf{W}_k \mathbf{U}_j$$

Relative Position Encoding

- Idea: relative positional encoding

$$\mathbf{A}_{i,j}^{\mathrm{abs}} = (\mathbf{E}_{x_i} + \mathbf{U}_i)^\top \mathbf{W}_q^\top \mathbf{W}_k (\mathbf{E}_{x_j} + \underbrace{\mathbf{U}_j}_{\text{absolute}})$$

$$\mathbf{A}_{i,j}^{\mathrm{rel}} = (\mathbf{E}_{x_i} + \mathbf{U}_i)^\top \mathbf{W}_q^\top \mathbf{W}_k (\mathbf{E}_{x_j} + \underbrace{\mathbf{R}_{i-j}}_{\text{relative}})$$

$$\mathbf{A}_{i,j}^{\mathrm{rel}} = \mathbf{E}_{x_i}^\top \mathbf{W}_q^\top \mathbf{W}_k \mathbf{E}_{x_j} + \mathbf{E}_{x_i}^\top \mathbf{W}_q^\top \mathbf{W}_k \mathbf{R}_{i-j} + \mathbf{U}_i^\top \mathbf{W}_q^\top \mathbf{W}_k \mathbf{E}_{x_j} + \mathbf{U}_i^\top \mathbf{W}_q^\top \mathbf{W}_k \mathbf{R}_{i-j}$$

$$\mathbf{A}_{i,j}^{\mathrm{rel}} = \mathbf{E}_{x_i}^\top \mathbf{W}_q^\top \mathbf{W}_{k,\underbrace{E}_{\text{content}}} \mathbf{E}_{x_j} + \mathbf{E}_{x_i}^\top \mathbf{W}_q^\top \mathbf{W}_{k,\underbrace{R}_{\text{position}}} \mathbf{R}_{i-j} + \mathbf{U}_i^\top \mathbf{W}_q^\top \mathbf{W}_{k,\underbrace{E}_{\text{content}}} \mathbf{E}_{x_j} + \mathbf{U}_i^\top \mathbf{W}_q^\top \mathbf{W}_{k,\underbrace{R}_{\text{position}}} \mathbf{R}_{i-j}$$

$$\mathbf{A}_{i,j}^{\mathrm{rel}} = \mathbf{E}_{x_i}^\top \mathbf{W}_q^\top \mathbf{W}_{k,E} \mathbf{E}_{x_j} + \mathbf{E}_{x_i}^\top \mathbf{W}_q^\top \mathbf{W}_{k,R} \mathbf{R}_{i-j} + \underbrace{u}_{\text{trainable}}{}^\top \mathbf{W}_{k,E} \mathbf{E}_{x_j} + \underbrace{v}_{\text{trainable}}{}^\top \mathbf{W}_{k,R} \mathbf{R}_{i-j}$$

  - the query vector is the same for all query positions
  - the attentive bias towards different words should remain the same

Relative Positional Encoding

- Idea: relative positional encoding
  - learnable embeddings → fixed embeddings with learnable transformations

absolute absolute

$$\mathbf{A}^{\mathrm{abs}}_{i,j} = \underbrace{\mathbf{E}_{x_i}^\top \mathbf{W}_q^\top \mathbf{W}_k \mathbf{E}_{x_j}}_{(a)} + \underbrace{\mathbf{E}_{x_i}^\top \mathbf{W}_q^\top \mathbf{W}_k \mathbf{U}_j}_{(b)} + \underbrace{\mathbf{U}_i^\top \mathbf{W}_q^\top \mathbf{W}_k \mathbf{E}_{x_j}}_{(c)} + \underbrace{\mathbf{U}_i^\top \mathbf{W}_q^\top \mathbf{W}_k \mathbf{U}_j}_{(d)}.$$

content position content position

$$\mathbf{A}^{\mathrm{rel}}_{i,j} = \underbrace{\mathbf{E}_{x_i}^\top \mathbf{W}_q^\top \mathbf{W}_{k,E} \mathbf{E}_{x_j}}_{(a)} + \underbrace{\mathbf{E}_{x_i}^\top \mathbf{W}_q^\top \mathbf{W}_{k,R} \mathbf{R}_{i-j}}_{(b)} + \underbrace{u^\top \mathbf{W}_{k,E} \mathbf{E}_{x_j}}_{(c)} + \underbrace{v^\top \mathbf{W}_{k,R} \mathbf{R}_{i-j}}_{(d)}.$$

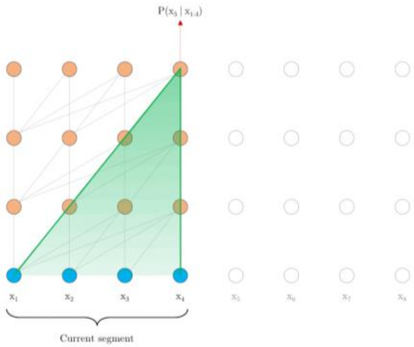
relative trainable parameters trainable parameters relative

much longer effective contexts than a vanilla model during evaluation

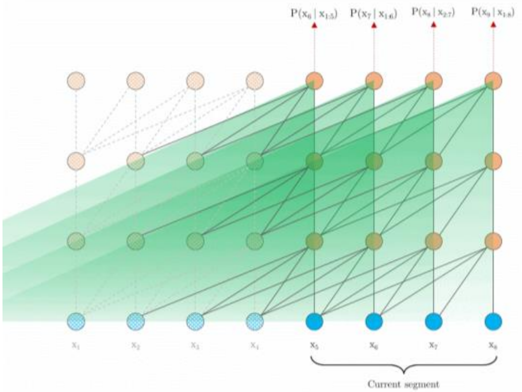
better generalizability to longer sequences

Segment-Level Recurrence in Inference

- Vanilla



- State Reuse



Contributions

- Longer context dependency
  - Learn dependency about 80% longer than RNNs and
    450% longer than vanilla Transformers
  - Better perplexity on long sequences
  - Better perplexity on short sequences by addressing the fragmentation issue
- Speed increase
  - Process new segments without recomputation
  - Achieve up to 1,800+ times faster than a vanilla Transformer during evaluation on LM tasks

13

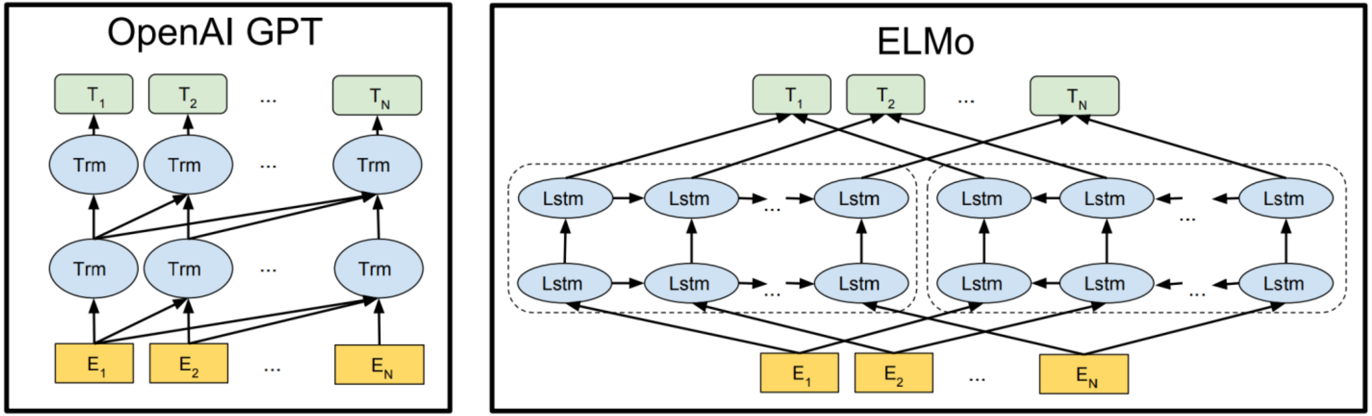
XLNet

(Yang et al., 2019)

Auto-Regressive (AR)

- Objective: modeling information based on either previous or following contexts

$$\max_{\theta} \quad \log p_{\theta}(\mathbf{x}) = \sum_{t=1}^{T} \log p_{\theta}(x_t \mid \mathbf{x}_{<t}) = \sum_{t=1}^{T} \log \frac{\exp\left(h_{\theta}(\mathbf{x}_{1:t-1})^{\top} e(x_t)\right)}{\sum_{x'} \exp\left(h_{\theta}(\mathbf{x}_{1:t-1})^{\top} e(x')\right)}$$

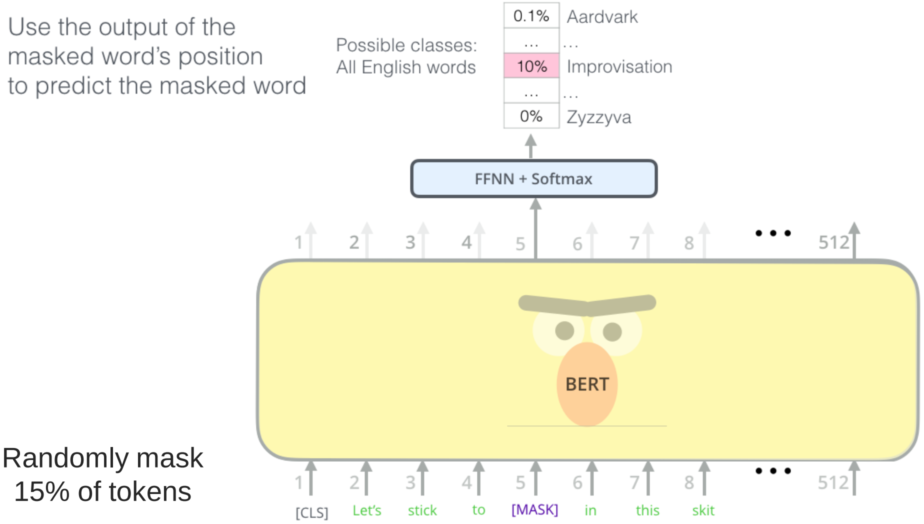


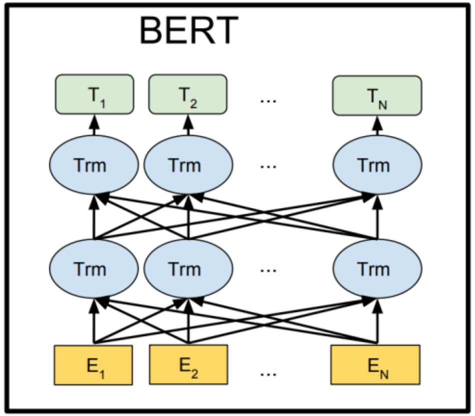
Auto-Encoding (AE)

- Objective: reconstructing $\bar{x}$ from $\hat{x}$

$$\max_{\theta} \quad \log p_{\theta}(\bar{\mathbf{x}} \mid \hat{\mathbf{x}}) \approx \sum_{t=1}^{T} m_t \log p_{\theta}(x_t \mid \hat{\mathbf{x}}) = \sum_{t=1}^{T} m_t \log \frac{\exp\left(H_{\theta}(\hat{\mathbf{x}})_t^{\top} e(x_t)\right)}{\sum_{x'} \exp\left(H_{\theta}(\hat{\mathbf{x}})_t^{\top} e(x')\right)}$$

  - dimension reduction or denoising (masked LM)





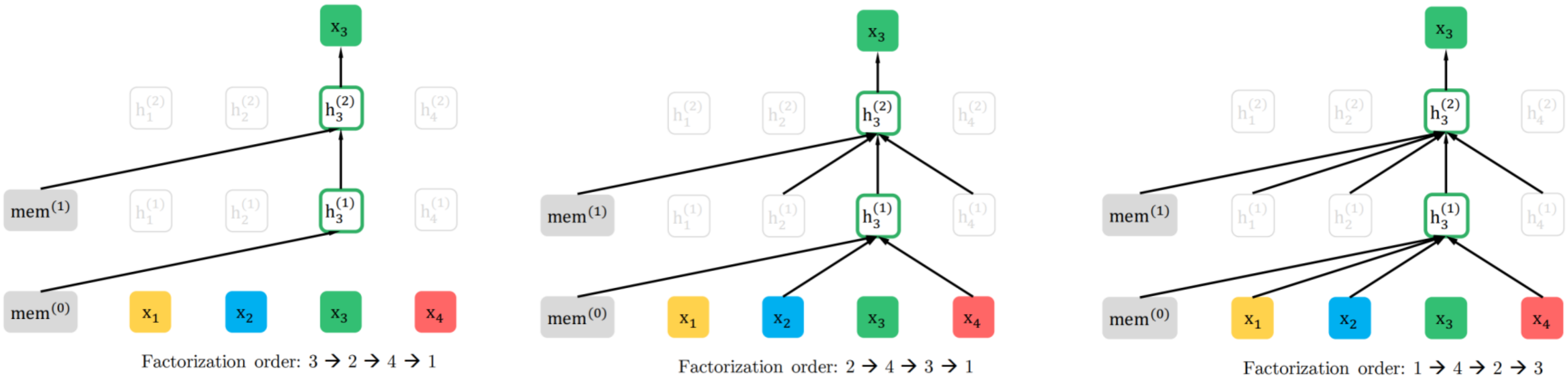
Auto-Encoding (AE)

- Issues
  - *Independence assumption*: ignore the dependency between masks
  - *Input noise*: discrepancy between pre-training and fine-tuning

    (w/ [MASK]) (w/o [MASK])

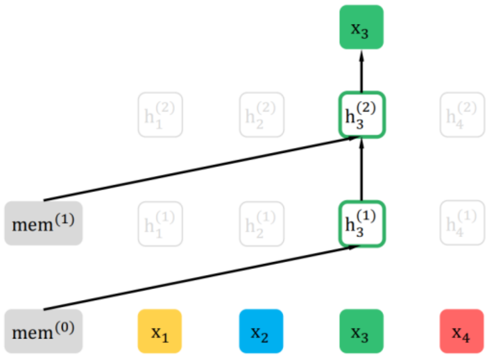
Permutation Language Model

- Goal: use AR and bidirectional contexts for prediction
- Idea: parameters shared across all factorization orders in expectation
  - T! different orders to a valid AR factorization for a sequence of length T
  - Pre-training on sequences sampled from all possible permutations



Permutation Language Model

- Implementation: only permute the factorization order
  - Remain original positional encoding
  - Rely on proper attention masks in Transformers



Factorization order: 3 → 2 → 4 → 1

	x_1	x_4	x_2	x_3
x_1	●			
x_4	●	●		
x_2	●	●	●	
x_3	●	●	●	●

	x_1	x_2	x_3	x_4
x_1	●			
x_2	●	●		●
x_3	●	●	●	●
x_4	●			●

resolve independence assumption and pretrain-finetune discrepancy issues

Formulation Reparameterizing

- Issue: naively applying permutation LM does not work
- Original formulation

$$p_\theta(X_{z_t} = x \mid \mathbf{x}_{z_{<t}}) = \frac{\exp\left(e(x)^\top \boxed{h_\theta(\mathbf{x}_{\mathbf{z}_{<t}})}\right)}{\sum_{x'} \exp\left(e(x')^\top \boxed{h_\theta(\mathbf{x}_{\mathbf{z}_{<t}})}\right)}$$

  - `[MASK]` indicates the target position
  - $h_\theta(x_{z_{<t}})$ does not depend on predicted position

$x_1, x_2, x_3, x_4 \rightarrow P(x_3 | x_1, x_2)$

$x_1, x_2, x_4, x_3 \rightarrow P(x_4 | x_1, x_2)$

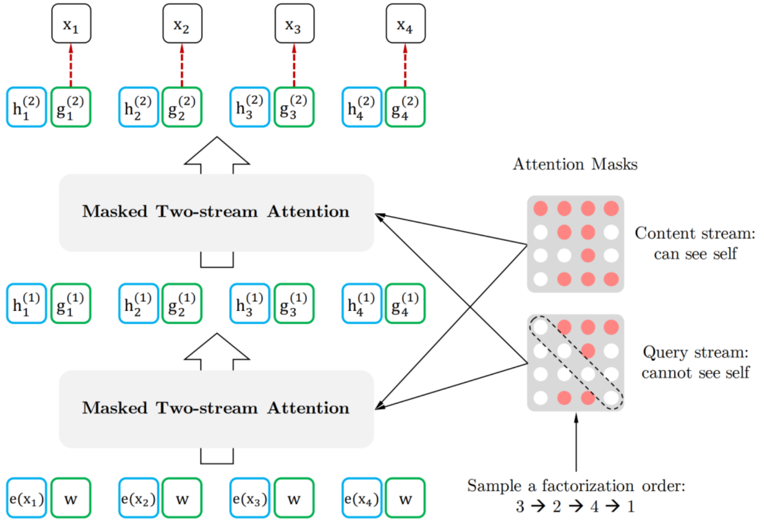
- Reparameterization

$$p_\theta(X_{z_t} = x \mid \mathbf{x}_{z_{<t}}) = \frac{\exp\left(e(x)^\top \boxed{g_\theta(\mathbf{x}_{\mathbf{z}_{<t}}, z_t)}\right)}{\sum_{x'} \exp\left(e(x')^\top \boxed{g_\theta(\mathbf{x}_{\mathbf{z}_{<t}}, z_t)}\right)}$$

  - $g_\theta(x_{z_{<t}}, z_t)$ is a new embedding considering the target position z_t

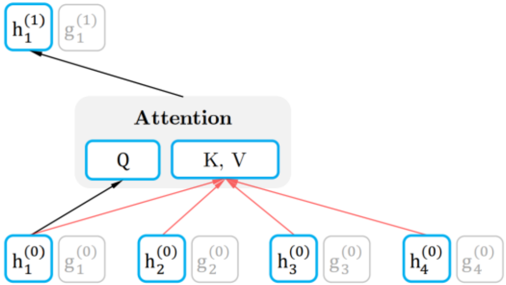
Two-Stream Self-Attention

- Formulation of $g(x_{z_{<t}}, z_t)$
  1) Predicting the token x_{z_t} should only use the position z_t and not the content x_{z_t}
  2) Predicting other tokens x_{z_j} $(j > t)$ should encode the content x_{z_t}
- Idea: two sets of hidden representations
  - Content stream: can see self
  - Query stream: cannot see self

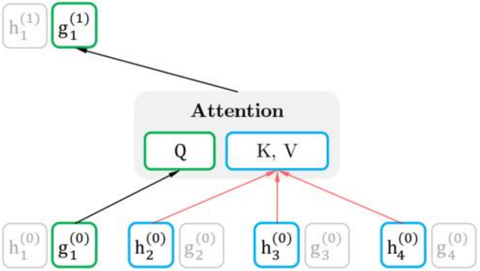


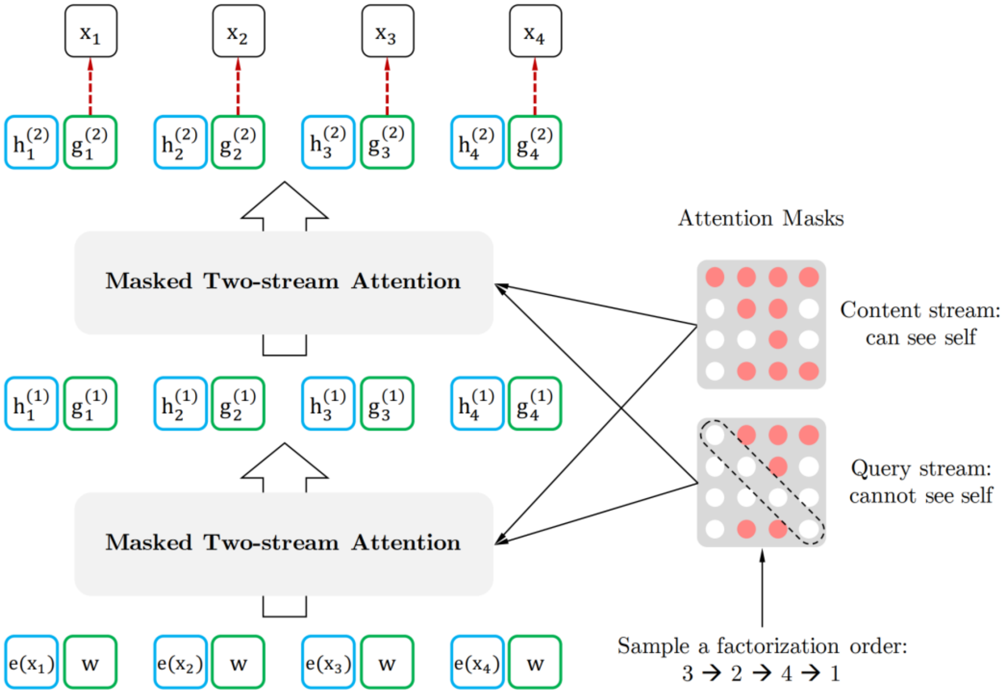
Two-Stream Self-Attention

- Content stream
  - Predict other tokens



- Query stream
  - Predict the current token





GLUE Results

Model	MNLI	QNLI	QQP	RTE	SST-2	MRPC	CoLA	STS-B	WNLI
Single-task single models on dev									
BERT [2]	86.6/-	92.3	91.3	70.4	93.2	88.0	60.6	90.0	-
XLNet	**89.8/-**	**93.9**	**91.8**	**83.8**	**95.6**	**89.2**	**63.6**	**91.8**	-
Single-task single models on test									
BERT [10]	86.7/85.9	91.1	89.3	70.1	94.9	89.3	60.5	87.6	65.1
Multi-task ensembles on test (from leaderboard as of June 19, 2019)									
Snorkel* [29]	87.6/87.2	93.9	89.9	80.9	96.2	91.5	63.8	90.1	65.1
ALICE*	88.2/87.9	95.7	**90.7**	83.5	95.2	92.6	**68.6**	91.1	80.8
MT-DNN* [18]	87.9/87.4	96.0	89.9	**86.3**	96.5	92.7	68.4	91.1	89.0
XLNet*	**90.2/89.7**†	**98.6**†	90.3†	**86.3**	**96.8**†	**93.0**	67.8	**91.6**	**90.4**

Contributions

- AR for addressing independence assumption

$$\mathcal{J}_{\text{BERT}} = \log p(\text{New} \mid \text{is a city}) + \log p(\text{York} \mid \text{is a city})$$

$$\mathcal{J}_{\text{XLNet}} = \log p(\text{New} \mid \text{is a city}) + \log p(\text{York} \mid \text{New}, \text{is a city})$$

- AE for addressing the pretrain-finetune discrepancy

$$\mathcal{J}_{\text{BERT}} = \sum_{x \in \mathcal{T}} \log p(x \mid \mathcal{N}); \quad \mathcal{J}_{\text{XLNet}} = \sum_{x \in \mathcal{T}} \log p(x \mid \mathcal{N} \cup \mathcal{T}_{<x})$$

24

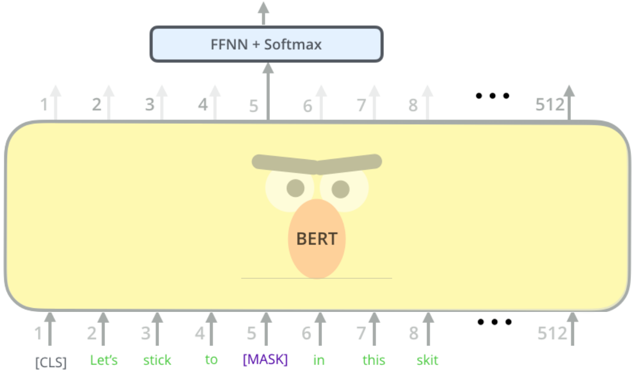
RoBERTa (Liu et al., 2019)

<u>R</u>obustly <u>o</u>ptimized <u>BERT</u> <u>a</u>pproach

What's More in RoBERTa

- Dynamic masking
  - 10 different masking ways over 40 epochs
    - BERT: static masking by preprocessing

Masking	SQuAD 2.0	MNLI-m	SST-2
static	78.3	84.3	92.5
dynamic	78.7	84.0	92.9



- Optimization
  - peak learning-rate & #warmup-steps tuned separately
  - large batch (batch size=8K)

batch size	learning rate	epochs	steps	perplexity	MNLI-m	SST-2
256	1e-4	32	1M	3.99	84.7	92.5
2K	7e-4	32	125K	3.68	85.2	93.1
		64	250K	3.59	85.3	**94.1**
		128	500K	3.51	85.4	93.5
8K	1e-3	32	31K	3.77	84.4	93.2
		64	63K	3.60	85.3	93.5
		128	125K	**3.50**	**85.8**	**94.1**

What's More in RoBERTa

- Data
  - train only with full-length sequences
    - BERT: on the reduced length
  - BookCorpus + English Wikipedia (16G), CC-News (76G), OpenWebText (38G), Stories (31G)

Model	**data**	**batch size**	**steps**	**SQuAD (v1.1/2.0)**	**MNLI-m**	**SST-2**
RoBERTa						
with BOOKS + WIKI	16GB	8K	100K	93.6/87.3	89.0	95.3
+ additional data (§3.2)	160GB	8K	100K	94.0/87.7	89.3	95.6
+ pretrain longer	160GB	8K	300K	94.4/88.7	90.0	96.1
+ pretrain even longer	160GB	8K	500K	**94.6/89.4**	**90.2**	**96.4**
$\text{BERT}_{\text{LARGE}}$						
with BOOKS + WIKI	13GB	256	1M	90.9/81.8	86.6	93.7
$\text{XLNet}_{\text{LARGE}}$						
with BOOKS + WIKI	13GB	256	1M	94.0/87.8	88.4	94.4
+ additional data	126GB	2K	500K	94.5/88.8	89.8	95.6

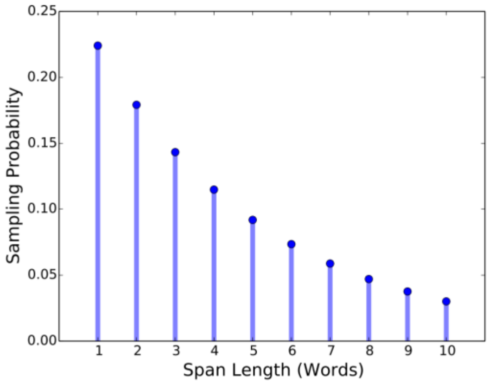
GLUE Results

	MNLI	QNLI	QQP	RTE	SST	MRPC	CoLA	STS	WNLI	Avg
Single-task single models on dev										
$\text{BERT}_{\text{LARGE}}$	86.6/-	92.3	91.3	70.4	93.2	88.0	60.6	90.0	-	-
$\text{XLNet}_{\text{LARGE}}$	89.8/-	93.9	91.8	83.8	95.6	89.2	63.6	91.8	-	-
RoBERTa	**90.2/90.2**	**94.7**	**92.2**	**86.6**	**96.4**	**90.9**	**68.0**	**92.4**	**91.3**	-
Ensembles on test (from leaderboard as of July 25, 2019)										
ALICE	88.2/87.9	95.7	**90.7**	83.5	95.2	92.6	**68.6**	91.1	80.8	86.3
MT-DNN	87.9/87.4	96.0	89.9	86.3	96.5	92.7	68.4	91.1	89.0	87.6
XLNet	90.2/89.8	98.6	90.3	86.3	**96.8**	**93.0**	67.8	91.6	**90.4**	88.4
RoBERTa	**90.8/90.2**	**98.9**	90.2	**88.2**	96.7	92.3	67.8	**92.2**	89.0	**88.5**

SpanBERT

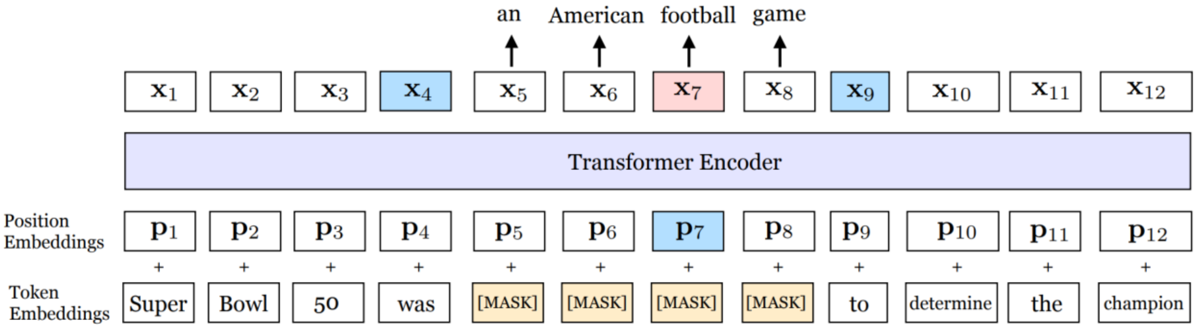
(Joshi et al., 2019)

SpanBERT



- Span masking
  - A random process to mask spans of tokens
- Single sentence training
  - a single contiguous segment of text for each training sample (instead of two)
- Span boundary objective (SBO)
  - predict the entire masked span using only the span's boundary

$$\mathcal{L}(\text{football}) = \mathcal{L}_{\text{MLM}}(\mathbf{x}_7) + \mathcal{L}_{\text{SBO}}(\mathbf{x}_4, \mathbf{x}_9, \mathbf{p}_7)$$



Results

- Masking scheme

	SQuAD 2.0	NewsQA	TriviaQA	Coreference	MNLI-m	QNLI
Subword Tokens	83.8	72.0	76.3	**77.7**	86.7	92.5
Whole Words	84.3	72.8	77.1	76.6	86.3	92.8
Named Entities	84.8	72.7	78.7	75.6	86.0	93.1
Noun Phrases	85.0	**73.0**	77.7	76.7	86.5	93.2
Random Spans	**85.4**	**73.0**	**78.8**	76.4	**87.0**	**93.3**

- Auxiliary objective

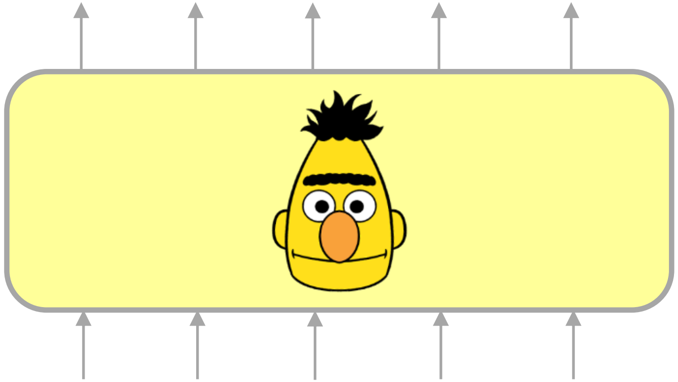
	SQuAD 2.0	NewsQA	TriviaQA	Coreference	MNLI-m	QNLI
Span Masking (2seq) + NSP	85.4	73.0	78.8	76.4	87.0	93.3
Span Masking (1seq)	86.7	73.4	80.0	76.3	87.3	93.8
Span Masking (1seq) + SBO	**86.8**	**74.1**	**80.3**	**79.0**	**87.6**	**93.9**

Multilingual BERT

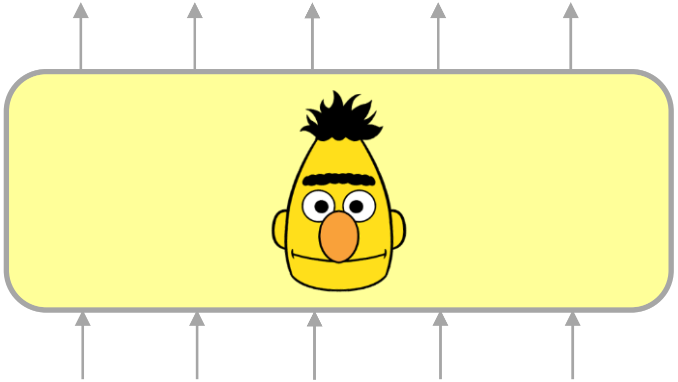
(Devlin et al., 2018)

Multilingual BERT

- Data: Wikipedia in top 104 languages
  - Code-mixing helps align words in different languages



《名偵探柯南》（日語：名探偵コナン），是日本漫畫家青山剛昌筆下的著名推理漫畫作品...



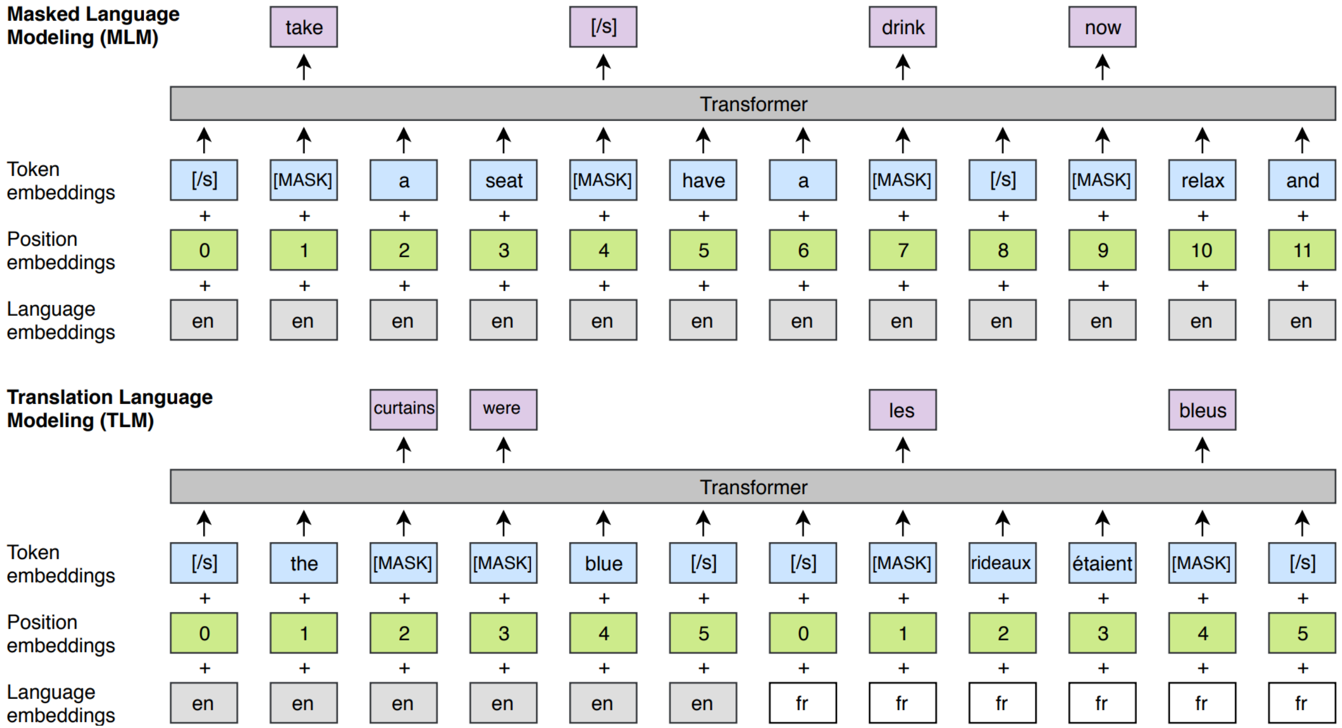
Case Closed, also known as Detective Conan (Japanese: 名探偵コナン, Hepburn: Meitantei Konan, lit. "Great Detective Conan"), is a Japanese detective manga series

XLM

(Lample & Connueau, 2019)

XLM

- Masked LM + Translation LM



Results

- Cross-lingual classification

	en	fr	es	de	el	bg	ru	tr	ar	vi	th	zh	hi	sw	ur	Δ
Machine translation baselines (TRANSLATE-TRAIN)																
Devlin et al. (2018)	81.9	-	77.8	75.9	-	-	-	-	70.7	-	-	76.6	-	-	61.6	-
XLM (MLM+TLM)	85.0	80.2	80.8	80.3	78.1	79.3	78.1	74.7	76.5	76.6	75.5	78.6	72.3	70.9	63.2	76.7
Machine translation baselines (TRANSLATE-TEST)																
Devlin et al. (2018)	81.4	-	74.9	74.4	-	-	-	-	70.4	-	-	70.1	-	-	62.1	-
XLM (MLM+TLM)	85.0	79.0	79.5	78.1	77.8	77.6	75.5	73.7	73.7	70.8	70.4	73.6	69.0	64.7	65.1	74.2
Evaluation of cross-lingual sentence encoders																
Conneau et al. (2018b)	73.7	67.7	68.7	67.7	68.9	67.9	65.4	64.2	64.8	66.4	64.1	65.8	64.1	55.7	58.4	65.6
Devlin et al. (2018)	81.4	-	74.3	70.5	-	-	-	-	62.1	-	-	63.8	-	-	58.3	-
Artetxe and Schwenk (2018)	73.9	71.9	72.9	72.6	73.1	74.2	71.5	69.7	71.4	72.0	69.2	71.4	65.5	62.2	61.0	70.2
XLM (MLM)	83.2	76.5	76.3	74.2	73.1	74.0	73.1	67.8	68.5	71.2	69.2	71.9	65.7	64.6	63.4	71.5
XLM (MLM+TLM)	**85.0**	**78.7**	**78.9**	**77.8**	**76.6**	**77.4**	**75.3**	**72.5**	**73.1**	**76.1**	**73.2**	**76.5**	**69.6**	**68.4**	**67.3**	**75.1**

Concluding Remarks

- Transformer-XL (https://github.com/kimiyoung/transformer-xl)
  - Longer context dependency
- XLNet (https://github.com/zihangdai/xlnet)
  - AR + AE
  - No pretrain-finetune discrepancy
- RoBERTa (http://github.com/pytorch/fairseq)
  - Optimization details & data
- SpanBERT
  - Better for QA, NLI, coreference
- Multilingual BERT (https://github.com/google-research/bert)
- XLM (https://github.com/facebookresearch/XLM)
  - Zero-shot scenarios